Gradient-based inference of abstract task representations for generalization in neural networks

Ali Hummos

Brain and Cognitive Sciences Department Massachusetts Institute of Technology Cambridge, MA ahummos@MIT.edu

Felipe del Río

Department of Computer Science Pontificia Universidad Catolica de Chile Santiago, Chile

Brabeeba Mien Wang

Computer Science and Artificial Intelligence Laboratory Massachusetts Institute of Technology Cambridge, MA

Julio Hurtado

Centre for Applications of Mathematical & Computing Sciences
University of Warwick

Cristian B. Calderon

Centro Nacional de Inteligencia Artificial Santiago, Chile

Guangyu Robert Yang

Brain and Cognitive Sciences Department Massachusetts Institute of Technology Cambridge, MA

Abstract

Humans and many animals show remarkably adaptive behavior and can respond differently to the same input depending on their internal goals. The brain not only represents the intermediate abstractions needed to perform a computation but also actively maintains a representation of the computation itself (task abstraction). Such separation of the computation and its abstraction is associated with faster learning, flexible decision-making, and broad generalization capacity. We investigate if such benefits might extend to neural networks trained with task abstractions. For such benefits to emerge, one needs a task inference mechanism that possesses two crucial abilities: First, the ability to infer abstract task representations when no longer explicitly provided (task inference), and second, manipulate task representations to adapt to novel problems (task recomposition). To tackle this, we cast task inference as an optimization problem from a variational inference perspective and ground our approach in an expectation-maximization framework. We show that gradients backpropagated through a neural network to a task representation layer are an efficient heuristic to infer current task demands, a process we refer to as gradientbased inference (GBI). Further iterative optimization of the task representation layer allows for recomposing abstractions to adapt to novel situations. Using a toy example, a novel image classifier, and a language model, we demonstrate that GBI provides higher learning efficiency and generalization to novel tasks and limits forgetting. Moreover, we show that GBI has unique advantages such as preserving information for uncertainty estimation and detecting out-of-distribution samples.

1 Introduction

Cognitive science and neuroscience hold a prominent place for (top-down) abstract task representations in many accounts of advanced cognitive functions in animals, including humans (Niv, 2019). The brain not only represents the intermediate abstractions needed to perform a task but also actively maintains a representation of the task itself (i.e., task abstraction, (Mante et al., 2013; Rikhye et al., 2018; Zhou et al., 2019; Vaidya et al., 2021; Hummos et al., 2022)). Such separation of the computation and its abstraction is theorized to support learning efficiency, as well as adaptive behavior. First, regarding learning efficiency, task abstractions facilitate learning by organizing experiences collected (Yu et al., 2021). Human participants learn faster once they discover the underlying task structure (Badre et al., 2010; Collins, 2017; Vaidya et al., 2021; Castañón et al., 2021). In fact, humans have a bias to discover and use such latent task structures even when they do not exist or might hurt their performance (Gaissmaier & Schooler, 2008; Collins, 2017). Second, regarding adaptive behavior, previous work has shown that the brain updates these task abstractions to meet current task demands of the environment and (re)composes them to solve new problems (Miller & Cohen, 2001; Collins & Koechlin, 2012). Indeed, depending on the context, the brain is able to respond to the same sensory input in novel ways by composing an appropriate task abstraction that guides processing of the input (Rikhye et al., 2018; Tafazoli et al., 2024), supporting flexible decision making and generalization to novel situations (Collins & Koechlin, 2012; Vaidya et al., 2021).

Traditional artificial neural networks (ANNs) architectures have a static computational graph that entangles input processing with task inference. For task-dependent computations in ANNs, one popular solution is to build in task representations in the networks. One solution is to build task representations into the structure of the network by using modular networks; where each module could represent a given task (Andreas et al., 2016; Kirsch et al., 2018; Goyal et al., 2019). Alternatively, one could regularize the weight updates such as to maximize an orthogonal computational space for each task (Kirkpatrick et al., 2017; Masse et al., 2018). Our framework is most related to models that add task encoding input layers, which provide information about the current task demands to the network (Yang et al., 2019; Hurtado et al., 2021; Wang & Zhang, 2022).

However, these previous models lack two important task inference features which we unpack in what follows and motivate our work. First, these models do not propose a mechanism to efficiently identify these tasks if they appear again in data, and thereby flexibly (re)adapt to the demands of previously encountered task¹. Second, and perhaps most importantly, these models lack a principled way of recomposing previously learned task abstractions, thereby minimizing the necessity for new learning (i.e. parameters update) to adapt to new situations (Lake et al., 2017).

To address the challenge of an efficient inference mechanism that can capture the two properties just described, we propose Gradient-Based Inference (GBI) of task abstractions, a solution grounded in variational inference, offering a robust and flexible framework. While traditional variational models have employed a static encoder to map data to an approximate posterior over latent representations (e.g. (Kingma & Welling, 2013)), they also have the potential to iteratively refine the posterior through rounds of optimization along gradients through the decoder (Rezende et al., 2014). Previous models have used iterative optimization in latent space to discover latent tasks and associate them with an internally generated latent task embedding (Butz et al., 2019; Hummos, 2023). While these models were able to generalize and learn continually, the iterative refinement can be computationally demanding. GBI tackles the computational efficiency by investigating a role for the decoder gradients in lieu of likelihoods and reformulating them in probabilistic terms. We show that one-step gradient updates can provide an informative distribution that has sufficient accuracy and can support detecting OOD samples and estimating uncertainty.

Our findings confirm that empirical findings from cognitive science do indeed extend to neural networks. In experiment 1 (toy dataset, section 3.1, we show that ANN models provided with abstract task representations do in fact show faster learning (improved data-efficiency), improved generalization to novel data points, and reduced forgetting. Importantly, we also confirm that a neural network trained with these abstractions has all the information to infer them at test time through back-propagated gradients, acting as a generative model that can estimate the likelihood of each abstraction and how well it explains the input data. In experiment 2 (image classification,

¹This work sidesteps the challenge of unsupervised discovery of task abstractions, a problem tackled by several algorithms (Butz et al., 2019; Xie et al., 2021; Mazzaglia et al., 2022; Hummos, 2023)

section 3.2), we first demonstrate that GBI can be used to perform image classification in a image generation setting. Second, we show that on-step gradient updates show interesting properties such as conveying uncertainty information and allowing for the detection of out-of-distribution (OOD) samples. Third, we consider a larger dataset with more categories using CIFAR-100. Finally, in experiment 3 (language experiment, section 3.3), we show that GBI is a domain-general method, and demonstrate several GBI properties (i.e., task inference, data efficiency, generalization) in language modeling.

2 Methods

Overview. We assume a dataset is equipped with a task abstraction. Conceptually, a task abstraction groups data samples into conceptual categories defined across dataset samples rather than individual data points. Mathematically, we model this by a graphical model where the data point \mathbf{X} is generated from a task abstraction \mathbf{Z} . Now we use a neural network to model this graphical model with data \mathbf{X} , task abstraction \mathbf{Z} and unknown parameter θ , our neural network model has the form of a joint likelihood function

$$\mathcal{L}(\theta; \mathbf{X}, \mathbf{Z}) = p(\mathbf{X}, \mathbf{Z}|\theta). \tag{1}$$

At the training phase, given a data point X with the abstraction of interest \hat{Z} directly observed, we train the neural network by doing gradient descent on $-\log \mathcal{L}(\theta; X, \hat{Z})$ with respect to θ .

At the test phase, we no longer assume access to task abstraction \mathbf{Z} and require the model to identify them in individual data points efficiently. Moreover, we require for our model a mechanism to manipulate the abstract representations to adapt to new unseen situations. Specifically, we update our \mathbf{Z} through gradient descent on $-\log \mathcal{L}(\theta; \mathbf{X}, \mathbf{Z})$.

The prediction of the model with task abstraction \mathbf{Z} is

$$\hat{\mathbf{X}} = \arg \max_{\mathbf{X}} \mathcal{L}(\theta; \mathbf{X}, \mathbf{Z}). \tag{2}$$

Connections to Expectation-Maximization algorithm. Notice that this method has deep connections to the classical Expectation-Maxmimization (EM) algorithm. In the training phase, since $\hat{\mathbf{Z}}$ is drawn from the distribution $p_{\mathbf{Z}|\mathbf{X}}$, the training attempts to find $\hat{\theta}$ such that

$$\hat{\theta} = \arg\max_{\theta} \mathbb{E}_{p_{\mathbf{Z}|\mathbf{X}}}[\log \mathcal{L}(\theta; \mathbf{X}, \mathbf{Z})]. \tag{3}$$

Equation 3 is a compact way to write the classical EM algorithm where the expectation steps are replaced by direct observation. Let the probability distribution of \mathbf{Z} be q. We can write

$$\log p(\mathbf{X}|\theta) = \mathbb{E}_q[\log p(\mathbf{X}|\theta)] \tag{4}$$

$$= \mathbb{E}_{q}[\log p(\mathbf{X}, \mathbf{Z}|\theta) - \log p(\mathbf{Z}|\mathbf{X}, \theta)]. \tag{5}$$

By adding and subtracting an entropy of q, we have

$$= \mathbb{E}_q[\log p(\mathbf{X}, \mathbf{Z}|\theta)] + H(q) + KL(q||p_{\mathbf{Z}|\mathbf{X}})$$
(6)

where KL divergence KL(q||p) is defined as $-\mathbb{E}_q[\log \frac{p}{q}]$ and the entropy H(q) is defined as $-\mathbb{E}_q[\log q]$. Now the expectation step corresponds to find q such that $\mathbb{E}_q[\log p(\mathbf{X}, \mathbf{Z}|\theta)] + H(q)$ is maximized. Notice that this is equivalent to minimizing the KL divergence $KL(q||p_{\mathbf{Z}|\mathbf{X}})$ and we know it is minimized at 0 when $q = p_{\mathbf{Z}|\mathbf{X}}$. In particular, we have $\mathbb{E}_q[\log p(\mathbf{X}, \mathbf{Z}|\theta)] + H(q) = \log p(\mathbf{X}|\theta)$.

At the maximization step, we find θ such that $\mathbb{E}_q[\log p(\mathbf{X}, \mathbf{Z}|\theta)] + H(q)$ is maximized, but since H(q) does not depend on θ , it is equivalent to maximize $\mathbb{E}_q[\log p(\mathbf{X}, \mathbf{Z}|\theta)]$ and therefore it is exactly Equation 3.

At our testing phase, since we are not able to access $p_{\mathbf{Z}|\mathbf{X}}$ directly, we optimize q through gradient descent. Our objective is exactly the expectation step with the regularization term H(q) implemented as L2 regularization on \mathbf{Z} .

There are different methods to optimize \mathbf{Z} and we discuss two methods we use in this paper, iterative optimization and one-step gradient update below to obtain gradient-based inference (GBI) estimates over latent variable values.

Iterative optimization of ${\bf Z}$. One method for doing gradient descent on ${\bf Z}$ is to iteratively optimize ${\bf z}$ using the gradient $\partial {\cal L}/\partial {\bf z}$. We implement iterative optimization using Adam optimizer with a learning rate of 0.01 and L2 regularization scaled by 0.01. This method usually takes many iterations (Marino et al., 2018). Another method is to learn an update rule $f_{\phi}(\cdot)$ that converts the gradient directly into an inferred ${\bf z} \leftarrow f_{\phi}(\partial {\cal L}/\partial {\bf z})$ (Marino et al., 2018). This method allows for rapid inference during run time but requires training of the update rule (like other meta-learning methods).

One-step gradient update. We explore an alternative that requires only one pass through the model by updating the gradient only once. Without iteratively optimizing ${\bf Z}$ through gradient update, one-step gradient update usually heavily depends on the initialization: because if the initialization is far away from the optimal ${\bf Z}$, the local gradient information might not be informative of the global optimum. However, notice that in the expectation step, our goal is to find the probability distribution of ${\bf Z},q$, such that $\mathbb{E}_q[\log p({\bf X},{\bf Z}|\theta)]+H(q)$ is maximized. We can consider an alternative objective function $f_\alpha(q)=\mathbb{E}_q[\log p({\bf X},{\bf Z}|\theta)]+\alpha H(q)$ for some $\alpha>0$ and let q^*_α be the point that maximizes this alternative objective function. Since entropy is concave, by increasing α we make the function more concave and therefore the gradient is more informative of the actual optimum. Furthermore, q^*_α will also become closer to the maximal entropy point $\bar{q}=\arg\max_q H(q)$. Therefore we can choose our initialization point as the maximal entropy point and output \hat{q} using our one step gradient update rule:

$$\hat{q} = \operatorname{softmax}(\bar{q} + \nabla f_{\alpha}(\bar{q})). \tag{7}$$

Here, we use a softmax function to project q back to the space of distribution. We implement this update using the SGD optimizer with learning rate 1 and no momentum terms. We reset our initialization point to maximal entropy to obtain \hat{q} for the next data point.

Code available at https://anonymous.4open.science/r/neuralbayes-F06F/.

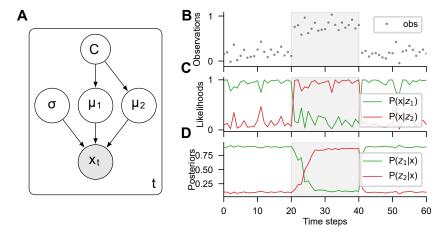


Figure 1: Sequence prediction task in one dimension. A) Bayesian graphical model generating the data. Task node (z) flips periodically and sets the mean of the Gaussian generating the data between $\mu_1 = 0.2$ or $\mu_2 = 0.8$, with fixed standard deviation σ at 0.1. B) Sample generated data observations. C) The likelihood of each context node value given data is calculated using the Gaussian probability density equation, more details in Appendix B. D) The posterior over z values.

3 Experiments

The experiments are organized as follows. In Section 3.1, we first investigate the properties of GBI on a simple synthetic dataset generated from a ground truth Bayesian model. We show that GBI displays better data-efficiency, generalizes better and forgets less, in addition to being able to pass neural gradients back to the Bayesian model to support Bayesian computations. Section 3.2 assesses the potential of GBI as a novel image classifier, the GBI ability to generate task samples, and to detect OOD samples. Finally, in Section 3.3, we demonstrate the abilities of GBI to recompose previously learned tasks, and thereby flexibly adapt to novel tasks using a language model.

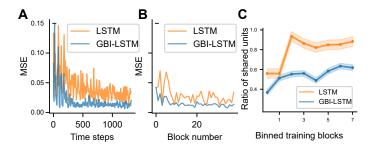


Figure 2: Comparing the training loss of an LSTM trained with task input (GBI-LSTM) without task input (LSTM). Note the variability in these loss estimates despite the simulation run over 10 random seeds is due to the stochastic block transitions still coinciding frequently. B) We show the mean loss per block but exclude the initial 20 predictions in each block to exclude poor performance from either model limited to transitioning between tasks. C) We quantify the ratio of shared neurons active in the 0.2 blocks and the 0.8 blocks using 'task variance' to identify engaged neurons. We binned training blocks into groups of 5 to aggregate the data and increase accuracy.

3.1 Experiment 1: The impact of task abstractions and the dynamics of GBI in a toy dataset

We begin with a simple dataset of one-dimensional observations unfolding in time, $\mathbf{x}_{0:t}$. The observations are generated from two alternating Gaussian distributions with distinct means ($\mathcal{N}(0.2, 0.1)$) or $\mathcal{N}(0.8, 0.1)$, Fig 1A, B). The ground truth generative causal model (Fig 1A) has the task node (\mathbf{z}) as a binary variable with values either $\mathbf{z}^1 = [0, 1]$ or $\mathbf{z}^2 = [1, 0]$. Context node switched between those two values with a probability $P_v = 0.005$, but we enforced a minimum block length of 20 and a maximum of 50 (further details in Appendix B). Knowing the ground truth generative model, we can analytically calculate the likelihood of each z^i as $p(x_t|z^i)$ and the posteriors $p(z^i|x_{0:t})$ (Fig 1C-E).

To estimate these Bayesian quantities from a neural network, we train a 100-unit LSTM (Hochreiter & Schmidhuber, 1997) to predict the next observation x_{t+1} given: (i) the five previous observations ($\{x_t, ..., x_{t-h}\}$, h = 5) presented sequentially, and (ii) a task abstraction input which we set to the task z values, one-hot encoded with either [0, 1] or [1, 0] after passing through a softmax function. We name this model GBI-LSTM and train it on the task (training details in appendix B), and in the following sections, we compare its learning dynamics to an LSTM with no such task input.

GBI-LSTM is data efficient. GBI-LSTM learns the task faster (i.e. better data efficiency) (Fig 2A,B). Example runs in appendix B (Fig S9), (LSTM, Fig S8). On one hand, this seems obvious as the GBI-LSTM gets additional contextual information, but on the other hand, the task is simple enough that it is not clear that a 100-unit LSTM needs the additional information. To explain, we reasoned that the available task abstraction deeply influences what solutions the network learns and allows for modules to emerge for each task. We used 'task variance' to identify neurons active in each task, a measure used in neuroscience literature (Yang et al., 2019). We identify active neurons by selecting units with the highest variance across samples of inputs and outputs for a task. We see that neurons for each task overlap in the LSTM but separate into modules in the GBI-LSTM, throughout training (Fig 2C). At *inference time*, with weights frozen, the baseline LSTM handles task transitions efficiently (Fig S8). The GBI-LSTM has the weights frozen but we now iteratively optimize the task abstraction layer z using vanilla SGD optimizer with a learning rate of 0.5 and we see that it also transitions between tasks by updating its task abstraction input z (Fig 3A-C, Fig S9).

GBI-LSTM generalizes better and forgets less. GBI-LSTM generalizes better to values outside of the training range (Fig 3D). By using iterative optimization, a gradient step in **Z** every time step, the GBI-LSTM can interpolate and better predict data points drawn from other distributions far from the two training distributions (Fig 3E). Moreover, we observe that the baseline LSTM already shows signs of catastrophic forgetting in this very simple setting. Testing MSE is worse around one of the two training means depending on which mean generated the last block of data during training. In figure 3D, we show the responses from runs where the last training block had the 0.2 Gaussian mean active. In contrast, as quantified in Table 1 the GBI-LSTM shows no signs such forgetting (Fig 3E).

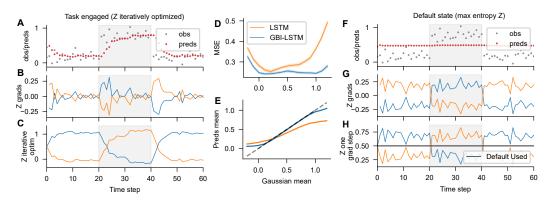


Figure 3: Neural network trained with task abstractions can dynamically tune itself to incoming data and generalizes to novel data points. A) Responses of the GBI-LSTM to sample data and B) the gradients of the prediction loss w.r.t. to the two Z units. C) Iteratively optimized Z values accumulating evidence dynamically. D) As a generalization test, while the network was trained on data from Gaussians with means 0.2 and 0.8, we test its responses to Gaussians between -0.2 to 1.2 (in 0.1 increments). We record the mean MSE values across 20 runs with different random seeds and compare to the baseline LSTM trained without task abstraction input. We only show runs where the last block of training was a 0.2 block, to highlight models behavior on that mean vs the other (0.8). E) The mean of the Gaussian data vs. the mean network predictions in each generalization block. These results are further quantified in Table 1. F-H) We show the other mode of using the same GBI-LSTM for gradient-based inference. We fix the task abstraction input Z to its state of maximum entropy (here [0.5, 0.5]) and take gradients from that point. While the network responds in the middle (F), the gradients w.r.t to Z (G) or similarly, the one-step gradient descent Z values behave much like the likelihood function, c.f. values computed using the Gaussian PDF in Fig 1C.

Table 1: Quantifying the generalization and forgetting on the toy task. Both models were trained on data generating means 0.2 and 0.8. We grouped models prediction errors on data generated from means -0.2 through 1.2 into four informative categories and evaluated mean MSE values \pm SEM. We identified the data generating mean during the last block of the training sequence, as the LSTM overfits to this mean and forgets data from the other mean. 20 runs with different seeds.

Data range	LSTM MSE	GBI-LSTM MSE
Mean of the last training block (0.2 or 0.8)	0.25 ± 0.02	0.24 ± 0.02
The other mean $(0.2 \text{ or } 0.8)$	0.30 ± 0.03	0.24 ± 0.02
Inside training range (0.3-0.7)	0.27 ± 0.02	0.25 ± 0.01
Outside training range ($<0.2 \& >0.8$)	0.35 ± 0.06	0.26 ± 0.03

Bayesian properties of GBI-LSTM. We can use our GBI-LSTM for inference, i.e. approximate Bayesian quantities such as the posterior and the likelihood function of z. We distinguish between two ways we can use the GBI-LSTM, first by taking one-step gradient updates from maximal entropy points ('default state' mode) and second is the iterative optimization of the task abstraction z at a lower learning rate ('task-engaged' mode) which we used above to perform the task and generalize to novel data. In the 'default state' mode, we set the task abstraction layer to its state of maximum entropy (here, the mean of the two values observed in training). For each time step, we take one step in z space to lower the prediction error, and then reset to max entropy for the next point. These one step z values (Fig 3H) behave much like the likelihoods of z computed through the ground truth causal model (Fig 1C). (See Fig S7 for a plot of the likelihoods and one gradient step z overlaid). Moreover, we see that these one step z can support Bayesian computations when used in lieu of the likelihood term in the Bayesian equation (previously calculated using the Gaussian probability distribution equation), i.e., we are able to pass one step z values to a Bayesian graphical model and Bayesian computations proceed normally (Fig S6). As such, the Bayesian model was able to offload learning parts of the causal graph to the neural network, but maintained the ability to pass belief updates and messages across the directed graph, despite the grafting of a neural network in the graph.

3.2 Experiment 2: GBI properties for image classification

We next explore the properties of gradient-based inference in more complex datasets. We trained a convolutional autoencoder to reconstruct MNIST digits. The decoder gets an additional input of a one-hot encoding of the digit label in addition to the latent embedding from the encoder (Fig 4 Schematic). We train this structure for 4 epochs using an MSE reconstruction loss (full details in Appendix C). Conforming the results observed in the toy task, the network receiving additional digit label input trains faster compared with baseline autoencoder, (Fig S12).

Gradients infer image class. At test time, we set the task abstraction (i.e., digit classes) input z to maximum entropy point and take one gradient step of the reconstruction loss with respect to the class inputs z, and directly interpret the resulting vector to infer the digit label. We make two kinds of comparison, first to methods that used

Table 2: GBI outperforms other gradient-based inference methods, and compares well to other canonical methods in ML (small drop in accuracy, but only one run through the neural network component). For iterative optimization, we further examine the relationship between optimization steps and accuracy in Fig S13.

<i>g</i>			
Method	Accuracy (%)	Runs	
Canonical methods			
Classifier	91.44 ± 0.51	1	
Pure generative	81.91 ± 2.3	10	
Ours			
GBI	85.46 ± 3.60	1	
Iterative Optimization	88.51 ± 3.42	50	
Likelihood (Recon MSE)	91.94 ± 3.38	10	
Other gradient-based methods			
EBI	27.37 ± 2.22	1	
NBI	78.78 ± 1.21	10	

gradient-based inference, we refer to them as entropy-based (Wortsman et al., 2020) and norm-based inference(Roy et al., 2024) and we see that GBI outperforms previous gradient-based methods (Table 2, implementation details in appendix C). Second, we compare GBI with four canonical methods to infer class based on the same architecture, as follows: (i) a convolutional classifier of the same architecture as the encoder, as a measure of what one might gain from a dedicated inference model that maps from inputs to task abstraction (in this case, digit class). (ii) iteratively optimizing the latent following the traditional variational inference paradigm (VI), which is the same as our model, but increases the computational budget by taking many smaller steps of gradient descent in task abstraction space. (iii) evaluating the model's MSE loss at each digit one-hot encoding and selecting the one with lowest MSE loss, as a general representation of Bayesian methods that require evaluating a likelihood function over all hypotheses of interest. (iv) evaluating the model's reconstruction MSE again, but in a purely generative model trained from task abstraction input to image reconstruction with no encoder, as an ablation to the separation of task input and task abstraction in our model. The goal: show that GBI suffers only a small drop in accuracy while offering a small computational budget and no added parameters. We see that GBI outperforms other gradient-based methods EBI and NBI, and show only a small drop in accuracy compared to canonical methods in machine learning (Table 2). We can also use our same model and increase the computational budget with iterative optimization or serial likelihood assessments when higher accuracy is required. More importantly, GBI offers a set of unique advantages as we discuss next.

Advantages of GBI in image classification. GBI retains many of the desirable properties of a generative model and thus can evaluate the probability of the data under the model. Therefore, we reasoned that GBI may have an advantage in uncertainty estimation (Fig S10) and in detecting out-ofdistribution (OOD) over traditional models. To test this, We compared a traditional convolutional classifier with GBI, both trained on MNIST digits and using fashionMNIST clothing items as the OOD dataset (Xiao et al., 2017). Using one gradient step task abstraction layer logits, GBI is superior at detecting OOD samples compared with the convolutional classifier (figure 4). In comparing our OOD detection results to current state-of-the-art methods we noted that the standard setup in studying OOD detection does not involve normalizing the image brightness mean and variance, making it trivial to discriminate the in and out-of-distribution datasets. We compared our OOD results to the Likelihood Regret (LR) method for OOD detection in a generative model Xiao et al. (2020) (see supplementary material for implementation details). On unnormalized data samples, where detection methods can take advantage of differences in pixel intensities between the in and out-of-distribution samples, both GBI and LR perform high (Fig A14, Table A6). However, when data samples are normalized GBI shows a clear advantage over LR (table 3). Lastly, we provide an illustration of how the GBI framework enables adaptive behavior, where in response to the same input (image), changing

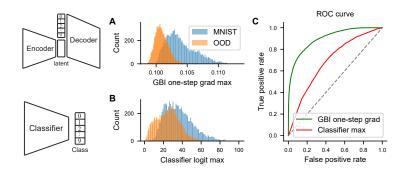


Figure 4: Robust out-of-distribution (OOD) samples detection in a GBI autoencoder compared to a classifier. Both models were trained on MNIST digits, and tested on OOD dataset fashionMNIST. We take the A) max value of the task abstraction layer in GBI and for the B) logits in the classifier and consider how are they distributed for in-distribution samples and OOD samples. C) ROC curves show that GBI max values are more discriminative.

the task abstraction (the digit label) can lead to a different input-to-output transformation, allowing for an additional layer of flexibility in task selection (Fig S11).

CIFAR-100. We apply GBI to the CIFAR-100 dataset, which features color images of vehicles, animals with diverse backgrounds. The GBI accuracy 18% well above that obtained from a pure generative model (Table 4) (i.e. a model to decode image class to image directly). This suggests a positive effect from the separation of visual input and task abstraction streams. We see only a small drop between evaluating the likelihood of each image class for each image (requires one for each of the 100 image classes, accuracy 21%) and GBI which requires only 1 run. Of note, we switched from an additive task abstraction to a multiplicative interaction to improve the accuracy, as backpropagation found solutions that favors relying on the visual input

Table 3: One gradient step values in GBI show better OOD detection compared to classifier logit and state-of-the-art method. AUCROC values averaged over 10 seeds with standard deviations reported. We examine the case when pixel intensity statistics were normalized so methods are not trivially detecting changes in brightness between the in-distribution and OOD datasets. Unnormalized results in Tab S6

 $\begin{tabular}{lll} \hline \textbf{Method} & \textbf{AUCROC} \\ \hline \textbf{GBI} & \textbf{0.89} \pm 0.03 \\ \hline \textbf{Classifier Softmax Maxes} & 0.73 \pm 0.08 \\ \hline \textbf{Likelihood Regret} & 0.80 \pm 0.06 \\ \hline \end{tabular}$

features rather than image class input (Table S8). This observation coupled with the overall low accuracy from the pure generative model suggests that image class input does not significantly reduce the variance seen in pixel space, leading the decoder to favor the visual features from the encoder. We anticipate that scaling to larger datasets will benefit from richer task abstractions. This highlights a role for algorithms that discover such abstractions from data (Butz et al., 2019; Hummos, 2023).

3.3 Experiment 3: Task-aware language modeling

Table 4: GBI accuracy on CIFAR100 using a multiplicative task abstraction. The task abstraction is projected through a frozen embedding layer (Bernoulli distribution) then multiplies the visual information going to decoder.

Method	Test Accuracy (%)	Model runs
Pure generative	9.57 ± 0.19	100
GBI	18.52 ± 0.38	1
Iterative optimization	18.53 ± 0.38	400
Likelihood	21.30 ± 0.36	100

We apply the concepts demonstrated in the Bayesian toy dataset and the MNIST classifier to language modelling for a self-supervised sequence modeling task. Specifically, we are interested in the effects of providing a task representation on data-efficiency, training dynamics and the ability to generalize, as observed in the case of the Bayesian toy dataset (Fig 1, 2, 3D,E).

Method We train an LSTM (with 2 layers, 200 units each) on word-level language prediction and compare an LSTM with our GBI-LSTM. To demonstrate the effectiveness of task abstrac-

tions in this setting, we represent each task abstraction as a one-hot encoded identifier of each training

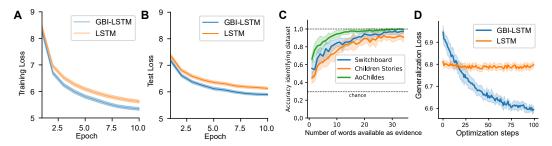


Figure 5: Better data-efficiency in an LSTM trained with task abstraction input (dataset ID one-hot) (GBI-LSTM) compared to training without such input (LSTM). Cross entropy loss for GBI-LSTM decreases faster for GBI-LSTM on the A) training set and B) testing set. Note that we train with a word-level tokenizer with a larger output dimension which has a highe cross entropy loss upper bound (12). Each simulation run had three language datasets picked randomly from 10 datasets. 10 data set picks, each run for 3 random seeds, shaded area SEM. C) One-pass gradient-based inference can infer dataset ID accurately at test time needing only a few words. GBI-LSTM takes a sequence of words available and outputs predictions over the same sequence shifted by one token (sequence to sequence modeling). We take the gradients w.r.t the task abstraction inputs and use the gradients to infer dataset ID. We show examples from 3 datasets to show variability. D) Iterative optimization can be used to adapt a GBI-LSTM to a novel dataset. We compare it to optimizing the inputs of an LSTM that was trained with no task provided to rule out any generic optimization effects. Due to the variability of loss values across datasets, for each dataset, we take the mean and SEM over 4 seeds and then aggregate the results across datasets.

dataset. We use datasets included in the BabyLM challenge (Warstadt et al., 2023), which provides 10 datasets inspired by what language data children might be exposed to during development, including wikipedia dataset, movie subtitles, children stories, and adult conversations, amongst others (listed in Appendix E). We train the two models on 3 of these datasets in a setting analogous to the Bayesian model dataset described earlier with data in each batch randomly drawn from one of the three datasets (we vary the choice of the 3 training datasets across seeds). The GBI-LSTM receives additional input with a one-hot encoded dataset identifier during training concatenated to the input token at each time step.

During testing, we use one-step gradients to infer the dataset from the text data provided as we assume no access to the dataset identifier at test time. We then use iterative optimization of the task abstraction input z to adapt the model to novel datasets by taking gradients steps on one batch of data and evaluating on the test set.

Results First, we focus on the effect of task abstractions on data-efficiency. Compared with the LSTM, GBI-LSTM displays lower training and validation losses (Fig 5A, B). While the improvement is modest, we emphasize that we only had to add <1.6 bits of information to obtain it.

Second, we focus on task inference. We show that GBI can quickly and reliably infer the dataset at hand, and in fact, it takes a few words from the first batch to accurately infer the dataset (Fig 5C). Note that similar to likelihood functions, gradient inference can handle variable amounts of evidence, and the estimation accuracy scales with the amount of evidence available.

Third, we focus on generalization. Our results show that task abstraction layers improves the model's ability to generalize to novel tasks (Fig 5D), similar to the case in the synthetic Bayesian dataset (Fig 3D,E). We use a 4th dataset not seen during training, and we compare the GBI-LSTM and LSTM losses on this novel dataset. To allow gradient descent to compose the closest solution to the novel datasets, we optimize the task abstraction input nodes of the GBI-LSTM to lower its prediction loss and we see a decrease in loss as it adjusts to this new unseen dataset (Fig 5D) (See Appendix E for example run). Notably, the GBI-LSTM starts out at a higher loss, on average, highlighting that task-aware models require task inference.

4 Conclusions and future directions

Overall we show that neural networks with access to task abstractions learn faster, forget less, adapt to changes in data distribution, and generalize to new situations. At the same time, we also provide tools for the networks to function independently infer their own task abstractions during testing. These task abstractions can be provided by humans, a trained teacher model, or inferred from data with no supervision (e.g. (Hummos, 2023)). Task-dependent networks require task inference and inferring an incorrect task impairs function. In animal brains, some computations that are frequently needed or are critical for survival might be better mapped to the default state of the brain. The brain shows distinct dynamics when not engaged in a specific task, the default mode, and we show previously unexplored connections between feedback signals in this default state and distributional inference signals that can support many probabilistic quantities such as uncertainty and surprise. We see future work addressing the limitation of relying on human-provided task abstractions as opposed to unsupervised discovery of richer multidimensional task representations along dimensions important for adaptation and generalization. Finally, while our method is theoretically motivated, we see future work extending these findings to larger datasets and more real-world applications.

References

- Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., and Nahavandi, S. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, December 2021. ISSN 1566-2535. doi: 10.1016/j.inffus.2021.05.008. URL https://www.sciencedirect.com/science/article/pii/S1566253521001081.
- Abdelali, A., Guzman, F., Sajjad, H., and Vogel, S. The AMARA Corpus: Building Parallel Language Resources for the Educational Domain. January 2014. doi: 10.13140/2.1.1806.2406.
- Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 39–48, 2016.
- Badre, D., Kayser, A. S., and D'Esposito, M. Frontal cortex and the discovery of abstract action rules. *Neuron*, 66(2):315–326, April 2010. ISSN 1097-4199. doi: 10.1016/j.neuron.2010.03.025.
- Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight Uncertainty in Neural Network. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1613–1622. PMLR, June 2015. URL https://proceedings.mlr.press/v37/blundell15.html. ISSN: 1938-7228.
- Butz, M. V., Bilkey, D., Humaidan, D., Knott, A., and Otte, S. Learning, planning, and control in a monolithic neural event inference architecture. *Neural Networks*, 117:135–144, September 2019. ISSN 0893-6080. doi: 10.1016/j.neunet.2019.05.001. URL https://www.sciencedirect.com/science/article/pii/S0893608019301339.
- Castañón, S. H., Cardoso-Leite, P., Altarelli, I., Green, C. S., Schrater, P., and Bavelier, D. A mixture of generative models strategy helps humans generalize across tasks. *bioRxiv*, pp. 2021.02.16.431506, January 2021. doi: 10.1101/2021.02.16.431506. URL http://biorxiv.org/content/early/2021/02/16/2021.02.16.431506.abstract.
- Collins, A. and Koechlin, E. Reasoning, Learning, and Creativity: Frontal Lobe Function and Human Decision-Making. *PLOS Biology*, 10(3):e1001293, March 2012. ISSN 1545-7885. doi: 10.1371/journal.pbio.1001293. URL https://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001293. Publisher: Public Library of Science.
- Collins, A. G. E. The Cost of Structure Learning. *Journal of Cognitive Neuroscience*, 29 (10):1646–1655, October 2017. ISSN 0898-929X. doi: 10.1162/jocn_a_01128. URL https://doi.org/10.1162/jocn_a_01128. _eprint: https://direct.mit.edu/jocn/article-pdf/29/10/1646/1953060/jocn_a_01128.pdf.
- Fetaya, E., Jacobsen, J.-H., Grathwohl, W., and Zemel, R. Understanding the Limitations of Conditional Generative Models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1lPleBFvH.

- Gaissmaier, W. and Schooler, L. J. The smart potential behind probability matching. *Cognition*, 109(3):416–422, December 2008. ISSN 0010-0277. doi: 10.1016/j.cognition.2008.09.007. URL https://www.sciencedirect.com/science/article/pii/S0010027708002151.
- Gal, Y. and Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1050–1059. PMLR, June 2016. URL https://proceedings.mlr.press/v48/gal16.html. ISSN: 1938-7228.
- Godfrey, J., Holliman, E., and McDaniel, J. SWITCHBOARD: telephone speech corpus for research and development. In [Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing, volume 1, pp. 517–520 vol.1, March 1992. doi: 10.1109/ICASSP. 1992.225858. ISSN: 1520-6149.
- Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., and Schölkopf, B. Recurrent Independent Mechanisms. *ArXiv*, 2019.
- Graves, A. Practical Variational Inference for Neural Networks. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://papers.nips.cc/paper_files/paper/2011/hash/7eb3c8be3d411e8ebfab08eba5f49632-Abstract.html.
- Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1321–1330. PMLR, July 2017. URL https://proceedings.mlr.press/v70/guo17a.html. ISSN: 2640-3498.
- Hendrycks, D. and Gimpel, K. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks, October 2018. URL http://arxiv.org/abs/1610.02136. arXiv:1610.02136 [cs].
- Hill, F., Bordes, A., Chopra, S., and Weston, J. The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations, April 2016. URL http://arxiv.org/abs/1511.02301. arXiv:1511.02301 [cs].
- Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.1997.9.8.1735. URL https://direct.mit.edu/neco/article/9/8/1735-1780/6109.
- Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., Laroussilhe, Q. D., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2790–2799. PMLR, May 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html. ISSN: 2640-3498.
- Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-Rank Adaptation of Large Language Models, October 2021. URL http://arxiv.org/abs/2106.09685. arXiv:2106.09685 [cs].
- Hummos, A. Thalamus: a brain-inspired algorithm for biologically-plausible continual learning and disentangled representations. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6orC5MvgPBK.
- Hummos, A., Wang, B. A., Drammis, S., Halassa, M. M., and Pleger, B. Thalamic regulation of frontal interactions in human cognitive flexibility. *PLOS Computational Biology*, 18(9):e1010500, September 2022. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1010500. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1010500. Publisher: Public Library of Science.
- Hurtado, J., Raymond, A., and Soto, A. Optimizing Reusable Knowledge for Continual Learning via Metalearning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 14150–14162. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/761e6675f9e54673cc778e7fdb2823d2-Abstract.html.

- Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An Introduction to Variational Methods for Graphical Models. In Jordan, M. I. (ed.), *Learning in Graphical Models*, pp. 105–161. Springer Netherlands, Dordrecht, 1998. ISBN 978-94-010-6104-9 978-94-011-5014-9. doi: 10.1007/ 978-94-011-5014-9_5. URL http://link.springer.com/10.1007/978-94-011-5014-9_ 5.
- Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. 2013. doi: 10.48550/ARXIV.1312. 6114. URL https://arxiv.org/abs/1312.6114. Publisher: arXiv Version Number: 11.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, March 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/10.1073/pnas.1611835114. Publisher: Proceedings of the National Academy of Sciences.
- Kirsch, L., Kunze, J., and Barber, D. Modular Networks: Learning to Decompose Neural Computation. November 2018. URL https://www.semanticscholar.org/paper/Modular-Networks%3A-Learning-to-Decompose-Neural-Kirsch-Kunze/0b50b9e103e19d87c2d30ed0d157d8379320ce6f.
- Lahiri, S. Complexity of Word Collocation Networks: A Preliminary Structural Analysis, March 2014. URL http://arxiv.org/abs/1310.5111. arXiv:1310.5111 [physics] version: 2.
- Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017. ISSN 0140-525X, 1469-1825. doi: 10.1017/S0140525X16001837. URL https://www.cambridge.org/core/product/identifier/S0140525X16001837/type/journal_article.
- Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.
- Lester, B., Al-Rfou, R., and Constant, N. The Power of Scale for Parameter-Efficient Prompt Tuning. Technical Report arXiv:2104.08691, arXiv, September 2021. URL http://arxiv.org/abs/2104.08691. arXiv:2104.08691 [cs] type: article.
- Li, X. L. and Liang, P. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353.
- Lison, P. and Tiedemann, J. OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 923–929, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1147.
- MacKay, D. J. C. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, May 1992. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.1992.4.3.448. URL https://direct.mit.edu/neco/article/4/3/448-472/5654.
- MacWhinney, B. *The CHILDES project: Tools for analyzing talk: Transcription format and programs, Vol. 1, 3rd ed.* The CHILDES project: Tools for analyzing talk: Transcription format and programs, Vol. 1, 3rd ed. Lawrence Erlbaum Associates Publishers, Mahwah, NJ, US, 2000. ISBN 0-8058-2995-4 (Hardcover). Pages: xi, 366.
- Mante, V., Sussillo, D., Shenoy, K. V., and Newsome, W. T. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, November 2013. ISSN 1476-4687. doi: 10.1038/nature12742. URL https://www.nature.com/articles/nature12742. Publisher: Nature Publishing Group.

- Marino, J., Yue, Y., and Mandt, S. Iterative Amortized Inference. *Proceedings of Machine Learning Research*, 80:3403–3412, July 2018. ISSN 1938-7228. URL https://resolver.caltech.edu/CaltechAUTHORS:20190506-143507942. Conference Name: 35th International Conference on Machine Learning Meeting Name: 35th International Conference on Machine Learning Place: Stockholm, Sweden Publisher: PMLR.
- Masse, N. Y., Grant, G. D., and Freedman, D. J. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44):E10467–E10475, October 2018. doi: 10.1073/pnas.1803839115. URL https://www.pnas.org/doi/10.1073/pnas.1803839115. Publisher: Proceedings of the National Academy of Sciences.
- Mazzaglia, P., Verbelen, T., Dhoedt, B., Lacoste, A., and Rajeswar, S. Choreographer: Learning and adapting skills in imagination. *arXiv preprint arXiv:2211.13350*, 2022.
- Miller, E. K. and Cohen, J. D. An integrative theory of prefrontal cortex function. *Annual Review of Neuroscience*, 24:167–202, 2001. ISSN 0147-006X. doi: 10.1146/annurev.neuro.24.1.167.
- Nalisnick, E., Matsukawa, A., Teh, Y. W., Gorur, D., and Lakshminarayanan, B. Do Deep Generative Models Know What They Don't Know? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1xwNhCcYm.
- Neal, R. M. and Hinton, G. E. A View of the Em Algorithm that Justifies Incremental, Sparse, and other Variants. In Jordan, M. I. (ed.), *Learning in Graphical Models*, NATO ASI Series, pp. 355–368. Springer Netherlands, Dordrecht, 1998. ISBN 978-94-011-5014-9. doi: 10.1007/978-94-011-5014-9_12. URL https://doi.org/10.1007/978-94-011-5014-9_12.
- Niv, Y. Learning task-state representations. *Nature Neuroscience*, 22(10):1544–1553, October 2019. ISSN 1546-1726. doi: 10.1038/s41593-019-0470-8. URL https://www.nature.com/articles/s41593-019-0470-8. Number: 10 Publisher: Nature Publishing Group.
- Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning*, pp. 1278–1286. PMLR, June 2014. URL https://proceedings.mlr.press/v32/rezende14.html. ISSN: 1938-7228.
- Rikhye, R. V., Gilra, A., and Halassa, M. M. Thalamic regulation of switching between cortical representations enables cognitive flexibility. *Nature Neuroscience*, 21(12):1753–1763, December 2018. ISSN 1546-1726. doi: 10.1038/s41593-018-0269-z.
- Roy, S., Verma, V., and Gupta, D. Efficient Expansion and Gradient Based Task Inference for Replay Free Incremental Learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1165–1175, January 2024.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56): 1929–1958, 2014. ISSN 1533-7928. URL http://jmlr.org/papers/v15/srivastava14a. html
- Tafazoli, S., Bouchacourt, F. M., Ardalan, A., Markov, N. T., Uchimura, M., Mattar, M. G., Daw, N. D., and Buschman, T. J. Building compositional tasks with shared neural subspaces. bioRxiv, 2024. doi: 10.1101/2024.01.31.578263. URL https://www.biorxiv.org/content/early/2024/03/22/2024.01.31.578263. Publisher: Cold Spring Harbor Laboratory _eprint: https://www.biorxiv.org/content/early/2024/03/22/2024.01.31.578263.full.pdf.
- Vaidya, A. R., Jones, H. M., Castillo, J., and Badre, D. Neural representation of abstract task structure during generalization. *eLife*, 10:e63226, March 2021. ISSN 2050-084X. doi: 10.7554/eLife.63226. URL https://doi.org/10.7554/eLife.63226. Publisher: eLife Sciences Publications, Ltd.
- Wang, D. and Zhang, S. Contextual Instance Decoupling for Robust Multi-Person Pose Estimation. pp. 11060-11068, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Wang_Contextual_Instance_Decoupling_for_Robust_Multi-Person_Pose_Estimation_CVPR_2022_paper.html.

- Warstadt, A., Choshen, L., Mueller, A., Williams, A., Wilcox, E., and Zhuang, C. Call for Papers The BabyLM Challenge: Sample-efficient pretraining on a developmentally plausible corpus, January 2023. URL https://arxiv.org/abs/2301.11796v1.
- Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J., and Farhadi, A. Supermasks in Superposition. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15173–15184. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/ad1f8bb9b51f023cdc80cf94bb615aa9-Paper.pdf.
- Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747. _eprint: 1708.07747.
- Xiao, Z., Yan, Q., and Amit, Y. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. *Advances in neural information processing systems*, 33:20685–20696, 2020.
- Xie, A., Harrison, J., and Finn, C. Deep Reinforcement Learning amidst Continual Structured Non-Stationarity. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11393–11403. PMLR, July 2021. URL https://proceedings.mlr.press/v139/xie21c.html.
- Yang, G. R., Joglekar, M. R., Song, H. F., Newsome, W. T., and Wang, X.-J. Task representations in neural networks trained to perform many cognitive tasks. *Nature Neuroscience*, 22(2):297-306, February 2019. ISSN 1546-1726. doi: 10.1038/s41593-018-0310-2. URL https://www.nature. com/articles/s41593-018-0310-2.
- Yu, L. Q., Wilson, R. C., and Nassar, M. R. Adaptive learning is structure learning in time. Neuroscience & Biobehavioral Reviews, 128:270–281, September 2021. ISSN 0149-7634. doi: 10.1016/j.neubiorev.2021.06.024. URL https://www.sciencedirect.com/science/article/pii/S0149763421002657.
- Zhou, J., Montesinos-Cartagena, M., Wikenheiser, A. M., Gardner, M. P. H., Niv, Y., and Schoenbaum, G. Complementary Task Structure Representations in Hippocampus and Orbitofrontal Cortex during an Odor Sequence Task. *Current Biology*, 29(20):3402–3409.e3, October 2019. ISSN 0960-9822. doi: 10.1016/j.cub.2019.08.040. URL https://www.sciencedirect.com/science/article/pii/S0960982219310905.

A Related Work

A.1 Adapting large models

Several methods have been developed for adapting large models to specific tasks without expensive full fine-tuning. Prompt and prefix tuning involve the use of learnable prompts (Lester et al., 2021) or a continuous prefix (Li & Liang, 2021) that guide the model's outputs. Lower-rank adaptation (Hu et al., 2021) modifies model parameters with a low-rank tensor, learned during adapting the model, while Adapters introduce small, task-specific modules within the network that are trained while the rest of the model remains fixed (Houlsby et al., 2019). Different from these methods, our approach to contextualizing a generative model proposes providing context signals or inferring them during the pretraining of the model. This allows the model to organize knowledge along these dimensions giving it an inductive bias anticipating variations along the same dimensions.

A.2 Iterative Variational Inference

Variational inference, a method that recasts inference as an optimization problem (Jordan et al., 1998; Neal & Hinton, 1998), has been instrumental to the development of deep variational models. However, recent work has primarily used the variational formulation for training, relying on fast amortized directed encoders to infer an approximate posterior distribution (Kingma & Welling, 2013; Rezende et al., 2014), thereby avoiding the potentially lengthy optimization loops of taking gradients through the decoder to refine the posterior distribution. To mitigate this, amortized variational inference employs an additional neural network that takes either the gradients from the decoder, or the reconstruction errors and updates the approximate posterior distribution with only a few iterations needed (Marino et al., 2018). Our work builds upon these insights, demonstrating that the initial gradient to the latent carries sufficient information to approximate likelihood functions. This reduces the need for multiple model runs or learning an additional model, but keeps iterative refinement as an option for complex samples.

A.3 Quantifying Uncertainty

Uncertainty quantification is a critical aspect of both Bayesian and neural network paradigms, with a rich and extensive history of literature, for review see (Abdar et al., 2021). Many approaches derive from the maximum likelihood interpretation of neural network training (MacKay, 1992), known as Bayesian Neural Networks (BNNs). Such models model uncertainty by placing prior distributions over the model's weights, with Variational Inference and dropout-based methods (like MCdropout and MCdropConnect) serving as practical approximations (Blundell et al., 2015; Graves, 2011; Gal & Ghahramani, 2016; Srivastava et al., 2014). Ensemble methods, such as Deep Ensembles, improve predictive performance and provide calibrated uncertainty measures by aggregating predictions of multiple independently trained neural networks (Lakshminarayanan et al., 2017). Lastly, temperature scaling is a post-processing method used to calibrate the model's softmax outputs, aligning the model's confidence with its prediction accuracy (Guo et al., 2017).

A.4 Out-of-Distribution (OOD) Detection

The role of out-of-distribution (OOD) detection is essential for the secure implementation of machine learning systems. This is due to the tendency of models to deliver erroneously confident predictions when faced with data that diverges from the distribution on which they were initially trained. Current OOD detection methods fall primarily into two classes: discriminator-based either the logit or the feature space (Hendrycks & Gimpel, 2018) or generation-based approaches which employ either the disparity in data reconstruction or the estimation of density in latent space (Nalisnick et al., 2019), with the appealing intuition that generative models capture the data distribution and can detect out of distribution samples. While one might suspect that our method works because it adds a generative objective, recent work showed that several families of generative models might assign a higher probability to out-of-distribution data than in-distribution ((Nalisnick et al., 2019; Fetaya et al., 2020)). Using their rich visual latent spaces, they can generalize easily to other datasets. Different that these models, our model separates computation of visual features into one stream and the task abstractions into another, instead of assessing OOD probability from the visual features, we selectively assess in the task abstraction space.

B Bayesian Toy Dataset

B.1 Bayesian graphical model task

B.1.1 Bayesian generative model

The task involves a sequence of observations x_t with a new observation at each time step t. The observations are generated from the following Bayesian model with context nodes z, a categorical variable, here 2-way (binary). Variable z is represented as one-hot encoded vector that selects the mean between two different configurations, while variance remains constant, as follows:

$$\mathbf{z}_0 = \text{random choice: } [0,1] \text{ or } [1,0] \tag{8}$$

with a transition probability matrix:

$$P(\mathbf{z}_t|\mathbf{z}_{t-1}) = \begin{bmatrix} 1 - p_v & p_v \\ p_v & 1 - p_v \end{bmatrix}$$
(9)

Where P_v (=0.05) is the volatility of the context, or the probability of context switching every time step. We enforce a minimum of 20 trials per block and a maximum of 50.

$$P(x_t|\mathbf{z}_t = [0,1]) = \mathcal{N}(0.8, \sigma)$$

$$P(x_t|\mathbf{z}_t = [1,0]) = \mathcal{N}(0.2, \sigma)$$
(10)

Then using Bayes rule we express the posterior over context units at newest time step \mathbf{z}_{t+1} given the history of the observations as follows:

$$P(z_{t+1}|x_{0:t+1}) = \frac{P(x_{t+1}|z_{t+1})P(z_{t+1}|z_{0:t}, x_{0:t})}{P(x_{t+1}|z_{0:t}, x_{0:t})}$$
(11)

and the likelihood function as the Gaussian probability distribution density:

$$P(x_t \mid z_t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_t - \mu_1)^2}{2\sigma^2}\right)$$
 (12)

B.1.2 Neural Model Architecture and Training

We employed the LSTM implementation provided by Pytorch, utilizing an input layer of size [1, 100] for mapping inputs to the LSTM, which consists of 100 hidden units. In the case of the GBI-LSTM the model received additional inputs from two additional units, the task abstraction units, after passing through a softmax activation function. An output layer of size [100, 2] was used to map to the output's mean and variance estimates. The model was trained to maximize the log-likelihood of the observed data by predicting a mean and a variance estimate, allowing the model to express a distribution over the next observation.. The final form of the objective derived from the Gaussian PDF has the MSE divided by the output variance. The LSTM was trained using the Adam optimizer with a learning rate of 10^{-3} . To optimize the context representations, we employed Adam with a learning rate of 10^{-3} .

C MNIST

C.1 Further results on MNIST and confidence

We compared the confidence values from the GBI autoencoder trained on MNIST digits to a convolutional classifier. The classifier and the encoder shared the same convolutional network, and the decoder used its transpose convolutions. Being interested in the errors the models make, we limited the capacity of the model and used a small number of convolutional filters, specifically we used these layers, (table 5):

The classifier used this backbone followed by a non-linear layer mapping to 10 digit classes, and then a softmax. While the GBI autoencoder produced the 8 dimensional embedding from the encoder, and concatenated the digit label one-hot vector and passed those to the decoder. Of importance, we

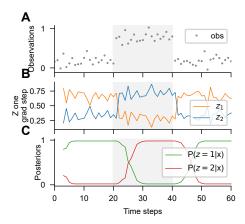


Figure 6: Grafting the neural gradients into Bayesian computations. With the task abstraction input z set to max entropy, we take one step along the gradients w.r.t to z. We use these one-step gradient updates in lieu of the likelihood function (eqn 12), thereby avoiding having to discover the μ and σ nodes of the underlying Bayesian causal model, and offloading the structure learning to the neural network, but maintaining the ability to proceed with Bayesian computations.

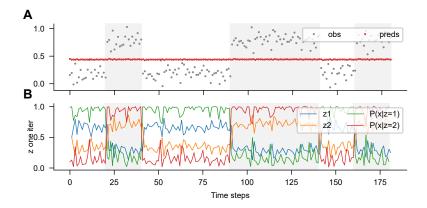


Figure 7: Overlaid likelihoods computed using the ground truth Bayesian model and neural gradients from GBI-LSTM for visual comparison. A) GBI-LSTM was in default mode with z at max entropy as responses and one step gradients recorded.

Table 5: Architecture of the convolutional classifier and encoder networks. Decoder had the same structure but transposed.

Layer	Input Channels	Output Channels	Stride	Activation
Conv2d	image_channels	2	2	ReLU
Conv2d	2	4	2	ReLU
Conv2d	4	8	-	None
Flatten	8	=	-	-

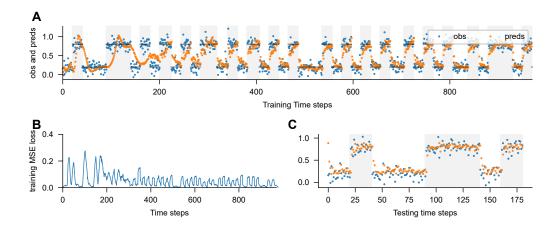


Figure 8: Training an LSTM on the Bayesian toy dataset, but no context from Bayesian nodes. An RNN trained on next observation prediction on this simple task learns it trivially well, and can adapt by detecting Gaussian distribution changes at context boundaries. Makes only one mistake typically before switching its predictions.

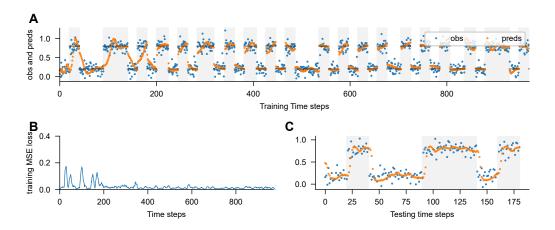


Figure 9: Training a GBI-LSTM on the Bayesian toy dataset with context from Bayesian model.

limited the encoder embedding output to only 8 dimensional vector, otherwise the decoder would ignore the labels and use the rich, but low level visual information from the encoder. We generated a few samples from the network by providing an original image from the test set, but changing the digit label from 0 through 9, and examined the conditionally generated images, to ensure that the decoder was using the digit label (Fig 11).

We examined the distribution of confidence values that GBI autoencoder produced, and found an informative distribution that can support post-hoc decision boundaries to get specific levels of accuracy, including 100% accuracy in the highest confidence bins. While a classifier trained with a softmax produces a distorted distribution that does not appear informative when binned (Fig S10). However, using the test set, one can post-hoc calibrate the softmax temperature of the classifier to obtain informative uncertainty estimates (Guo et al., 2017). Accordingly, our observation is that GBI does not require any post-hoc re-calibration to get an informative confidence distribution, but such as distribution is not unique.

Other gradient-based methods details. First methods is norm-based inference(Roy et al., 2024). These authors use the gradients norm to infer the task. They compute the gradient norm for each possible task label and select the one with the lowest norm. Second, we compared our method to entropy-based gradient inference (EBI), following the work of Wortsman et al. (2020). They run a

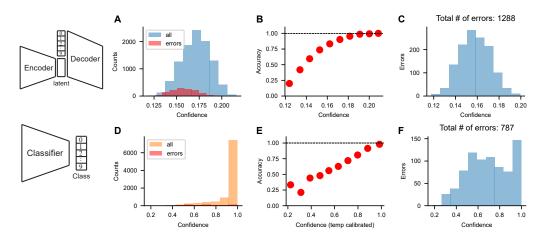


Figure 10: Confidence distributions and relationship with accuracy comparing a GBI network trained with task abstractions and a convolutional classifier.

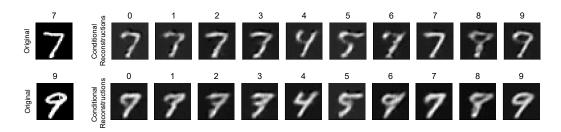


Figure 11: Conditional generation from the GBI autoencoder network. We feed an original image to the encoder and ask the decoder to reconstruct the image, as we vary the digit label input 0 to 9.

forward pass based on the superposition of all task labels each weighted by an α value. Subsequently, they compute the output layer activation entropy, and take a step in α such as to minimize that entropy. To perform a one-iteration inference, they select the supermask (or task) with the largest gradient to minimize the entropy. Given that this method is limited to cross-entropy loss we reframe image generation as minimizing a cross-entropy pixel-wise classification loss.

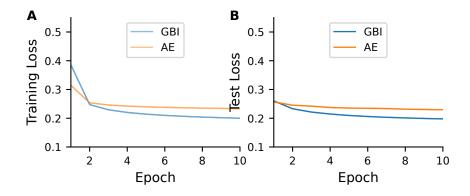


Figure 12: Comparing task-aware (GBI) and context-free autoencoder (AE) training and validation error on MNIST.

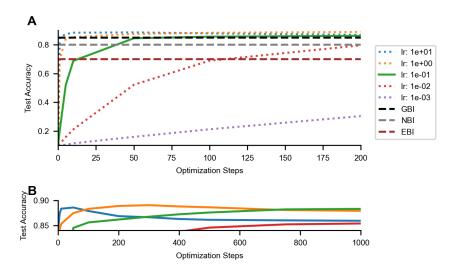


Figure 13: Variational inference iterative optimization steps vs accuracy. The models were randomly initialized in a classical variational inference fashion, and the latent representation was trained using vanilla SGD, which proved to be more stable than other optimization algorithms such as Adam and allowed control over learning rates. A) We highlight the accuracy line with the best end accuracy, other learning rates in dashed lines. Horizontal lines mark the three gradient-based methods compared in Table 2. B) A closer look at how higher learning rates eventually lead to lower accuracy, and we picked the best learning rate based on end accuracy being the higest.

	AUCROC (Unnormalized)
Method	Value
GBI Maxes	0.90 ± 0.05
Classifier Softmax Maxes	0.68 ± 0.03
Likelihood Regret	1.00 ± 0.00

Table 6: Same comparison as in table 3, but with the in-distribution and OOD datasets having different brightness statistics.

D OOD Detection

We provide the GBI gradient and classifier logit maximum values and ROC in the case where the OOD dataset was not normalized. i.e., the OOD images had a different image brightness and variance (Fig S14), but we argue that an ideal image system should ignore these broad changes in illumination rather than rely on them to detect OOD samples.

E Language Model

E.1 Further experimental details on training task-aware language model

We based our language modeling approach on the official PyTorch example for training a word-level language model, which can be found at https://github.com/pytorch/examples/tree/main/word_language_model. We largely adopted the original design choices and parameters, including the usage of a stochastic gradient descent optimizer with a decaying learning rate, a sequence length of 35 words and a batch size of 20, gradients clipped to a value of 0.25, input layer and output layer dropout rate 0.25, word-level tokenizer, and a log-softmax function at the final output layer. We used the LSTM implementation. The tokenizer's output was passed through an embedding layer, followed by concatenation with a latent representation, specifically a one-hot encoding of the dataset ID. The LSTM output passed through an MLP layer to project the LSTM's hidden units back to the number of tokens, and a log-softmax function was applied as the final output activation.

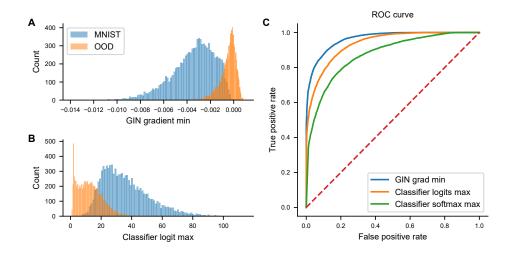


Figure 14: The figure compares the minimum gradient values of the (A) gradient based inference network (GBI) and the maximum classifier logits when fed an in-distribution image (MNIST) versus an out-of-distribution (OOD) image (fashionMNIST). B) The OOD images have a darker average intensity compared to MNIST, resulting in lower responses in the classifier logits. Exploiting this discrepancy, a discriminating threshold based on classifier logits yields a reasonable ROC curve (C) for distinguishing between the two distributions.

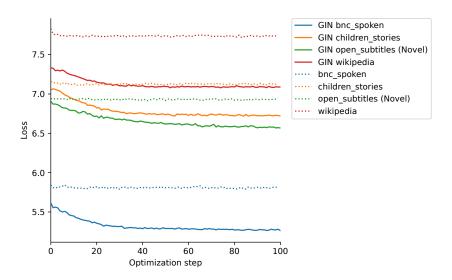


Figure 15: An example run showing the loss on predicting words from the same sequence while optimizing the latent context inputs

Dataset	Domain	Dataset Size
AoCHILDES (MacWhinney, 2000)	Child-directed speech	0.44M
British National Corpus (BNC), dialogue portion	Dialogue	0.86M
Children's Book Test (Hill et al., 2016)	Children's books	0.57M
Children's Stories Text Corpus ²	Children's books	0.34M
Standardized Project Gutenberg Corpus (Lahiri, 2014)	Written English	0.99M
OpenSubtitles (Lison & Tiedemann, 2016)	Movie subtitles	3.09M
QCRI Educational Domain Corpus (QED; (Abdelali et al., 2014))	Educational video subtitles	1.04M
Wikipedia ³	Wikipedia (English)	0.99M
Simple Wikipedia ⁴	Wikipedia (Simple English)	1.52M
Switchboard Dialog Act Corpus (Godfrey et al., 1992)	Dialogue	0.12M
Total	-	9.96M

Table 7: The datasets included in the BabyLM Challenge (Warstadt et al., 2023), please see the original paper for further details on datasets and sources. The authors reported the number of words in the training set of each corpus. http://www.natcorp.ox.ac.uk https://www.kaggle.com/datasets/edenbd/children-stories-text-corpus https://dumps.wikimedia.org/enwiki/20221220/ https://dumps.wikimedia.org/simplewiki/20221201/

Note that we trained the model on 3 datasets at a time and tested generalization on a fourth dataset to keep the tokenizer from indexing more than 1 million distinct words. Adding more datasets lead to the embedding layer having a massive size that would not fit in a GPU memory.

Here we provide brief information on the datasets included in the BaybLM challenge. We use their more challenging smaller "Strict-Small" data.

Method	Test Accuracy (%)	Model runs
GBI	10.92	1
Iterative optimization	12.77	400
Likelihood	14.88	100

Table 8: Test results for the performance of different methods in CIFAR100 using a concatenated one-hot task abstraction to the encoder output.